A Statistical Indoor Localization Method for Supporting Location-based Access Control

Article *in* Mobile Networks and Applications · March 2009

DOI: 10.1007/s11036-008-0143-4 · Source: DBLP

5 authors, including:

Zhen Yu
Wuhan University of Science and Technology
100 PUBLICATIONS **2,155** CITATIONS

SEE PROFILE

Steve F. Russell
Iowa State University
116 PUBLICATIONS **187** CITATIONS

SEE PROFILE

A Statistical Indoor Localization method for Supporting Location-based Access Control

by

Chunwang Gao

A thesis submitted to the graduate faculty

in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Major: Computer Engineering

Program of Study Committee:
Steve F. Russell, Major Professor
Yong Guan, Co-Major Professor
William Q. Meeker, Committee Member
Douglas W. Jacobson, Committee Member

Iowa State University

Ames, Iowa

2009

DEDICATION

I would like to dedicate this thesis to my wife Zheng and to my son William without whose love and support I would not have been able to complete this work.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ACKNOWLEDGEMENTS

I would like to to express my sincere thanks to those who helped me with various aspects of this research and the writing of this thesis. First and foremost, Dr. Russell for his years' of guidance and patience throughout my study and my research during the time I spent in and out of Iowa State University. His support and encouragement have often inspired me and renewed my hopes for completing this graduate education. I would like to thank Dr. Guan for his numerous support of this research by providing ideas, equipment and many precious discussion meetings. This certainly could not be done without your effort. I would also like to thank my committee members, Dr. Meeker and Dr. Jacobson, for their precious contributions to this work. I would also like to thank my officemates Zhen Yu, Linfeng Zhang, Yawen Wei for their kindness and lifetime friendship formed during the time I worked in Coover 3223. Finally, I would like make a special thank to Zhen Yu for his effort and help which made the publication of this work possible when I was out in Virginia.

CHAPTER 1. Introduction

Location awareness is critical for supporting location-based access control (LBAC). The challenge is how to determine locations accurately and efficiently in indoor environments. Existing solutions based on WLAN signal strength either cannot provide high accuracy, or are too complicated to accommodate to different indoor environments. In this thesis, we propose a statistical indoor localization method for supporting location-based access control. In an offline phase, we fit a locally weighted regression and smoothing scatterplots (LOESS) model on the signal strength received at different training locations, and build a radio map that contains the distribution of signal strength. In an online phase, we determine the locations of unknown points using maximum likelihood estimation (MLE) based on the measured signal strength and the stored distribution. In addition, we provide 95% confidence intervals to our estimation using Bootstrapping method. Compared with other approaches, for example, in [3] and [16], our method is simpler, more systematic and more accurate. Experimental results show that the estimation error of our method is less than $2m$. Hence, it can better support LBAC applications than others.

CHAPTER 2. A Statistical Indoor Localization method for Supporting Location-based Access Control

2.1 Introduction

Traditional access control systems identify and authenticate users based on something they know (e.g., password or passphrase), something they have (e.g., access token or crypto-card), or something they are (e.g., fingerprint or voice). However, none of these methods is perfect. Passwords may be guessed. Tokens can be stolen, and fingerprints are vulnerable to replay. Fortunately, the information of user location offers a new dimension for authentication and access control. For example, to grant an access to some service, we can require that a user be present at a specific location (e.g., in a room or office). Otherwise, the access is denied. It is called Location-based Access Control (LBAC) [2, 6, 7], which provides more reliable access control when combined with traditional methods. In addition, it offers the ability to trace an intruder back to a physical location if some intrusion has been detected.

Location awareness is critical for supporting location-based access control. The challenge is how to determine locations accurately and efficiently, especially in indoor environments. Existing indoor approaches utilize different types of signals such as infrared [15], ultrasound [10], and radio frequency (RF) [3, 16, 11, 12, 17, 13] to estimate locations. Among these approaches, the RF-based ones are the most promising, because they can be easily integrated with existing and widely spread 802.11 infrastructure. RF-based approaches can be further classified depending on the metrics they measure, such as Triangulation [12], Time of Arrival (ToA) [17], Time Difference of Arrival (TDoA) [13] and Received Signal Strength (RSS) [3, 16, 11]. Since measuring AoA, ToA and TDoA requires special hardware such as directional antennas or fine-grained timers, localization based on WLAN signal strength seems more attractive and

has become more popular, which is also our focus.

Indoor localization approaches using wireless LAN signal strength approach typically consist of two phases such as an offline training phase and an online localization phase. In the offline phase, the signal strength from (or received by) different access points at different locations is recorded and used to build a radio map. Then, in the online phase, the measured signal strength is compared with that stored in the radio map to find the best location signal-strength match and hence determine the corresponding location. RADAR [3] measures signal strength by averaging a number of samples received by several access points from a mobile client within a period of time. Horus [16] identifies different causes for signal strength variations, and proposes corresponding solutions to these variations. This approach makes the Horus system complicated and imposes the need to make adjustments for each specific indoor environment. Lim et al. [11] observed that the indoor environments are time-variant. Lim proposed using real-time measurements for addressing environment dynamics and hence the offline phase is unneeded in their approach. However, they adopted a simple linear model mapping between a signal strength and the logarithm of a distance. This simple linearization is the main source of localization errors in their approach.

In this paper, we propose a statistical indoor localization method using WLAN signal strength for supporting location-based access control. In an offline phase, we fit a LOESS [4, 5] local regression model on a training set to build a radio map, which stores the distribution of signal strength. In an online phase, we determine the location using Maximum Likelihood Estimation (MLE) [8] based on the measured signal strength and the distribution stored in the radio map. We further exploit a Bootstrapping method [9] to estimate the confidence intervals for our method. Compared with existing approaches, our method is simpler and more accurate which will be illustrated in the later section. It can be generalized for any indoor environments. Experimental results show that the estimation error of our method is less than $2m$, so it can provide better support to LBAC applications than other approaches.

The paper is organized as follows: Related work is introduced in section 2.2. In Section 2.3, we discuss each component of our method including LOESS model, MLE and Bootstrapping

modules in detail. Then, we present experimental results in section 2.4, and conclude in section 2.5.

2.2 Related work

RADAR [3] is a two-phase indoor localization system using WLAN signal strength. In the offline phase, three base stations measure the average signal strength from a mobile client and build a radio map recording locations, signal strength, and users' directions. In the online phase, it uses a K-nearest approach search a location in the radio map, which best matches the measured signal strength. RADAR has high location errors, because the simple average value cannot represent the variation of signal strength precisely. Moreover, RADAR cannot be accommodated to different mobile clients whose signal strength is different.

Horus [16] is also a two-phase localization system. Unlike RADAR, Horus stores in the radio map the distribution of signal strength collected by a mobile client from different access points and determines the location using Bayes' theorem. Horus identifies different causes of signal strength variation and proposes corresponding solutions. For example, it uses an autoregressive model to handle the correlation between different samples from the same access point and utilizes a perturbation technique to deal with small-scale variation of signal strength. To obtain a continuous location estimation, Horus exploits a time-average window to smooth the resulting location. It achieves a high accuracy, but it is too complicated and has many parameters that should be adjusted for different indoor environments. So, it is not systematic for general LBAC applications.

Lim et al. observed that the indoor environments are time-variant. which means that the environmental model learned in the offline phase may not be suitable for the data collected in the online phase. Thus, they proposed a Zero-Configuration system [11] which updates the environmental model continuously without an offline phase. However, this system assumes a simple linear relationship between a signal strength value and a distance. This assumption cannot capture the dynamic property of indoor environments accurately and become the main source of location error.

2.3 Our statistical method

2.3.1 Framework of location-based access control

We considered a simple application of location-based access control: Alice is an employee of the financial department of some company. She is allowed to connect to the company's server from her wireless laptop and manage a database that contains the salary information of all employees of the company. To gain access to the database (or the server), Alice must provide not only her password, but also her location information. The company's security policy requires that she be present in a particular office when managing the database. It protects the database by imposing both network access security and password privacy. Together, with other access security such as a RADIUS server, the database is made much more secure.For example, Bob, an attacker who manages to steal Alice's password, could potentially connect to the company network via wired or wireless access from anywhere on the company premises. However, if valid database access is restricted to that specific office and a locked door prevents his physical access, he cannot gain network access to the database.

Suppose that several access points in the company's buildings have been equipped with location-based access control. Further suppose that a location-based access control program has been downloaded from the company's server to Alice's laptop. This program can automatically measure the signal strength from the access points and send this measured information to the server, when Alice logs on. The problem becomes how we could design a method to determine Alice's location accurately based on the measured signal strength.

2.3.2 Overview of our method

We propose a statistical method to determine locations based on the signal strength of access points for supporting location-based access control. Statistical method has the advantage of effective use of data with well developed mathematical theory as background. Fig. 2.1 illustrates the structure of our method that consists of an offline and an online phases.

In the offline phase, we first measure the signal strength received from different access points at each known location. Then, the offline measurements are fitted into a LOESS local

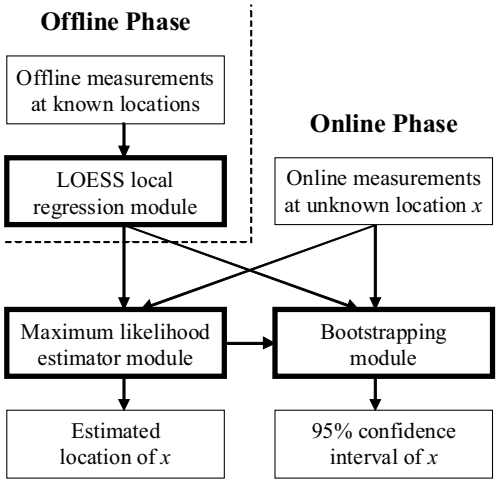


Figure 2.1 The structure of our statistical method

regression module, which builds a radio map containing the distribution of signal strength received at each location. In the online phase, the signal strength measured at an unknown location x is processed by a Maximum Likelihood Estimator (MLE) module based on the distribution obtained in the offline phase to generate an estimation of location x. Meanwhile, a Bootstrapping module outputs a 95% confidence interval for the estimated location.

Compared with existing solutions, our method has several advantages: (1) The LOESS module produces a model independent of any physical model. So, our method does not need to study the complicated theoretical model of signal strength in indoor environments. (2) The MLE module has the nice properties such as asymptotic normality and asymptotic unbiased minimum variance estimation. So, our estimation is theoretically robust, unlike Horus that needs to be adjusted for each different environment. (3) The Bootstrapping module provides a confidence interval for location estimation, which is more meaningful than just a single estimation value. (4) Our method is simpler, more efficient, and has a better location accuracy.

In the rest of the section, we discuss the LOESS, MLE and Bootstrapping modules in detail.

2.3.3 LOESS local regression module

2.3.3.1 Introduction to LOESS

The local regression fitting method (LOESS) [4, 5] was first proposed by Cleveland in 1979. It fits curves and surfaces to noisy data with locally weighted polynomial regression. A low degree polynomial is fitted to each point in the data set by giving more weight to nearby points and less to points farther away. The biggest advantage of LOESS is that it does not need to fit a specific function to all the samples. In addition, its flexibility and simplicity make it ideal for modeling very complex situations, when no clear theoretical model exists. This is the exact situation for the signal strength distribution inside a building.

2.3.3.2 Detailed procedure of LOESS

Given a data set of n points $\{x_1, y_1\}, \cdots, \{x_n, y_n\}$, the purpose of LOESS is to find a proper polynomial regression function g_i for each point $\{x_i, y_i\}$ such that

$$y_i = g_i(x_i) + \epsilon_i \ , \tag{2.1}$$

where ϵ_i is the regression error. (Note: In our localization method, x_i represents a location and y_i denotes the signal strength of some access point received at x_i.) The degree of the polynomial g_i is denoted by d, a pre-defined parameter. When $d = 1$, g_i is a function of straight line. When $d = 2$, y_i corresponds to a quadratic model, that is,

$$y_i = \beta_{i,0} + \beta_{i,1} x_i + \beta_{i,2} x_i^2 + \epsilon_i \ , \tag{2.2}$$

where $\beta_{i,0}$, $\beta_{i,1}$ and $\beta_{i,2}$ are estimated according to the optimization criterion specified by LOESS.

As its name, *local regression*, suggests, LOESS fits the regression function to each point $\{x_i, y_i\}$ using k ($k = nq$) points that are closest to $\{x_i, y_i\}$, where q is a smoothing parameter. Let $\{x_{min}, y_{min}\}$, $\cdots$, $\{x_i, y_i\}$, $\cdots$, $\{x_{max}, y_{max}\}$ denote these k closest points. Typically, we set

$$min = i - \lfloor \frac{k-1}{2} \rfloor \ \text{ and } \ max = i + \lfloor \frac{k-1}{2} \rfloor \ . \tag{2.3}$$

For example, when $i = 5$ and $k = 6$, we have $min = 2$ and $max = 7$, which mean that LOESS chooses points $\{x_2, y_2\}$, $\cdots$, $\{x_7, y_7\}$ to fit regression function g_5 to point $\{x_5, y_5\}$. Equation (2.3) is not applicable to the case that $min < 1$ or $max > n$, but we can easily find that the k nearest points should be $\{x_1, y_1\}$, $\cdots$, $\{x_k, y_k\}$, or $\{x_{n-k+1},\ y_{n-k+1}\}$, $\cdots$, $\{x_n, y_n\}$.

LOESS does not treat each of k nearest points equally. In fact, each point is assigned a weight depending on its distance to $\{x_i, y_i\}$. Let $d_{max} = \max(|x_j - x_i|)$ denote the maximum distance between x_i and x_j, for $x_j \in [x_{min}, x_{max}]$. The weight for the point at x_j is

$$w(x_j) = (1 - (\frac{|x_j - x_i|}{d_{max}})^3)^3 \ . \tag{2.4}$$

Considering the polynomial model shown in equation (2.2) and a weighted least-squares estimator, LOESS needs to estimate $\{\hat{\beta}_{i,0}, \hat{\beta}_{i,1}, \hat{\beta}_{i,2}\}$ that minimize the following quantity:

$$Q = \sum_{j=min}^{max} w(x_j)(y_j - (\beta_{i,0} + \beta_{i,1}x_j + \beta_{i,2}x_j^2))^2 \ . \tag{2.5}$$

The corresponding minimization criteria are,

$$\frac{\partial Q}{\partial \beta_{i,j}} = 0 \qquad \text{for} \quad j = 0, 1, 2. \tag{2.6}$$

This estimation can also be expressed in matrix. Let us define

$$Y = \begin{pmatrix} y_{min} \\ \vdots \\ y_{max} \end{pmatrix}, \qquad X = \begin{pmatrix} 1 & x_{min} & x_{min}^2 \\ \vdots & \vdots & \vdots \\ 1 & x_{max} & x_{max}^2 \end{pmatrix},$$

$$\vec{\beta} = \begin{pmatrix} \beta_{i,0} \\ \beta_{i,1} \\ \beta_{i,2} \end{pmatrix}, W = \begin{pmatrix} w(x_{min}) & & \\ & \ddots & \\ & & w(x_{max}) \end{pmatrix}.$$

The weighted least-squares estimator of $\vec{\beta}$ is:

$$\hat{\vec{\beta}} = (X^T W X)^{-1} X^T W Y \tag{2.7}$$

It can shown that the results calculated from equations (2.6) and (2.7) are equivalent. Given the estimation of $\vec{\beta}$ as

$$\hat{\vec{\beta}} = \begin{pmatrix} \hat{\beta_{i,0}} \\ \hat{\beta_{i,1}} \\ \hat{\beta_{i,2}} \end{pmatrix},$$

LOESS fits a quadratic model at point $\{x_i, y_i\}$ as

$$\hat{y}_i = g_i(x_i) = \hat{\beta}_{i,0} + \hat{\beta}_{i,1} x_i + \hat{\beta}_{i,2} x_i^2 \ . \tag{2.8}$$

The regression function $g_i(x_i)$ is calculated repeatedly at every point in the data set $\{x_1, y_1\}, \cdots, \{x_n, y_n\}$.

2.3.3.3 Choosing the smoothing parameter q

The most important two parameters controlling LOESS are d and q. Once we determine the value of d (e.g., $d = 2$), q is chosen from $[(d+1)/n, 1]$, which controls how much amount of data is used in each polynomial regression. However, which value of q is the best?

In our method, we determine the value of q by finding the model minimizing Akaike's Information Criterion (AIC) [1]. AIC is one of the most commonly used penalized model selection criteria. One version of the bias-corrected AIC value for a LOESS model might be

$$AIC_C = log(\hat{\sigma}^2) + 1 + \frac{2(Trace(L)+1)}{n - Trace(L) - 2} \ , \tag{2.9}$$

where n is the number of data points and $\hat{\sigma}$ is the standard error of data. $Trace(L)$ is the trace of matrix L, which is the smoothing matrix of the LOESS model. L defines the linear relationship between the fitted and observed dependent variable values. That is, L satisfies

$$\hat{Y} = LY \ , \tag{2.10}$$

where $\hat{Y}$ can be calculated using equation (2.8).

2.3.3.4 Fitting the LOESS model using the training set

Considering a system with m access points and n known test locations, we first collect the signal strength from all access points at each test location in an offline training phase. Our

training set is $\{x_1, s_{1,j}\}$, $\cdots$, $\{x_i, s_{i,j}\}$, $\cdots$, $\{x_n, s_{n,j}\}$, where x_i for $i = 1, \cdots, n$ denotes a test location and $s_{i,j}$ for $j = 1, \cdots, m$ denotes the signal strength of the j-th access point received at location x_i. The LOESS model on the training set can be expressed as

$$s_{i,j} = g_{i,j}(x_i) + \epsilon_{i,j} \ , \tag{2.11}$$

where $g_{i,j}$ and $\epsilon_{i,j}$ denote the regression function and regression error for the j-th access point at location x_i.

Then, we estimate $g_{i,j}$ based on the LOESS model and the chosen q. We assume that the regression error $\epsilon_{i,j}$ satisfies some normal distribution, that is,

$$\epsilon_{i,j} \sim \mathcal{N}(0, \ \sigma_{i,j}^2) \ , \tag{2.12}$$

where $\sigma_{i,j}^2$ denotes some variance. Thus, the signal strength also satisfies a normal distribution, that is,

$$s_{i,j} \sim \mathcal{N}(g_{i,j}(x_i), \ \sigma_{i,j}^2) \ . \tag{2.13}$$

This distribution is stored in the server and will be used in MLE module to determine locations.

2.3.4 Maximum likelihood estimator module

2.3.4.1 Introduction to MLE

Maximum Likelihood Estimation (MLE) [8] was formally proposed by Fisher. Given a large size of samples, MLE provides an unbiased minimum variance estimation and its estimates are approximately normally distributed.

Considering a probability distribution family with a probability density (mass) function f_θ, which is parameterized by unknown θ (a scalar or a vector). Given a set of observations $\{x_1, x_2, ..., x_k\}$ drawn from the distribution f_θ, the likelihood function for this set of observations is

$$L(\theta) = f_\theta(x_1, x_2, ..., x_n|\theta) \ . \tag{2.14}$$

If the observations are independent of each other, the likelihood function can be further written as

$$L(\theta) = \prod_{i=1}^{k} f_\theta(x_i|\theta) \ . \tag{2.15}$$

Applying logarithmic transformation to both sides of equation (2.15), the likelihood can also be expressed as

$$l(\theta) = \log\left(L(\theta)\right) = \sum_{i=1}^{k} \log\left(f_\theta(x_i|\theta)\right) \ . \tag{2.16}$$

The maximum likelihood estimator for θ, denoted as $\hat{\theta}$, is the value that maximizes the likelihood $L(\theta)$ or $l(\theta)$.

2.3.4.2 Location estimation using MLE

In the online phase, a user collects the signal strength $s_1, \cdots, s_j, \cdots, s_m$ from m access points at an unknown location x. The user sends the measured signal strength to the server, which then estimates location x using our MLE module and decides if an access should be granted to the user.

From the LOESS fitting result and the normal distribution in equation (2.13), the probability density function $f_j(s_j|x)$ for the signal strength s_j of the j-th access point received at location x is

$$f_j(s_j|x) = \phi_{\mathcal{N}}(\frac{s_j - g_j(x)}{\sigma_j}) \ , \tag{2.17}$$

where g_j and σ_j denote the regression function and the corresponding standard deviation of the j-th access point. $\phi_{\mathcal{N}}$ is the standard normal density function such that

$$\phi_{\mathcal{N}}(t) = \frac{1}{\sqrt{2\pi}} e^{-\frac{t^2}{2}} \ . \tag{2.18}$$

The likelihood function for the set of signal strength received from all of m access points at location x is

$$L(x) = \prod_{j=1}^{m} f_j(s_j|x) = \prod_{j=1}^{m} \phi_{\mathcal{N}}(\frac{s_j - g_j(x)}{\sigma_j}) \ . \tag{2.19}$$

In practice, we often calculate the logarithmical likelihood

$$l(x) = \log\left(L(x)\right) = \sum_{j=1}^{m} \log\left(\phi_{\mathcal{N}}(\frac{s_j - g_j(x)}{\sigma_j})\right) \ . \tag{2.20}$$

In our method, the server estimates location x using maximum likelihood estimator $\hat{x}$, which maximizes $L(x)$ or $l(x)$.

2.3.5 Bootstrapping module

Bootstrapping [9] is a statistical method for estimating the distribution of an estimator by re-sampling the original data. With this method, we can easily compute the confidence interval of estimation with higher accuracy than with other methods based on normal-approximation. In the online phase, we take advantage of a Bootstrapping module to give the confidence interval of our estimation, while neither of the approaches discussed previously can report such a confidence interval.

The procedure of Bootstrapping module consists of two steps. In step 1, the signal strength is re-sampled from the distribution $s_j(x) \sim \mathcal{N}(g_j(x), \sigma_j^2)$, which are calculated in the offline phase by the LOESS module, to get a new sampled training set. In step 2, a new estimator for x is calculated using MLE from the new training set. Iterating these two (i.e., re-sampling and estimating) steps usually several thousands times, we get a sample distribution for the new estimator $\hat{x}$, a 95% confidence interval is then calculated by selecting 2.5% and 97.5% quantile values as the lower and upper bounds.

2.4 Experimental results

We evaluated the performance of our method by experiments. Our experiments consisted of two parts. In the first part, we focused on one-dimensional estimation. In the second part, we tested our method in a two-dimensional scenario.

2.4.1 Setup for one-dimensional experiments

In our one-dimensional experiments, we collected the signal strength from three access points: (1) Netgear WGR614, (2) Linksys WRTSL54GS, and (3) IASTATE hot spot, which are deployed on the third floor of Coover Hall at Iowa State University. The floor is shaped like

a horizontally flipped "**Γ**" that consists of a $27.1m$ north-south corridor and a $30.8m$ east-west corridor. Fig. 2.2 shows the floor plan.

We placed the Linksys and Netgear access points at the two ends of corridors, while the IASTATE access point controlled by Iowa State University, was hung on the ceiling at the center of the floor. Along the middle line of the corridors, we selected 48 different points to measure signal strength, where the distance between the points was 1.23 meter. These points belong to two sets with equal size of elements. One is a training set used to build a radio map, and the other is a test set used for location estimation. The points were selected and placed into the two sets alternately. That is, the points of odd index such as 1, 3, 5, $\cdots$, 47 belong to the training set, while the test set contains the points of even index such as 2, 4, 6, $\cdots$, 48.

We measured the signal strength of different access points using a Dell Inspiron 8200 laptop, which was equipped with Windows XP, NetStumbler software, and a Linksys WUSB54GP external wireless adapter. NetStumbler is a tool for Windows that facilitates detection of Wireless LANs using the 802.11b, 802.11a and 802.11g WLAN standards. Fig. 2.3 shows a snapshot of NetStumbler. At each of the 48 points, the signal strength of every access point was measured every 0.5 second for one minute interval (i.e., 120 samples per access point). The median of these 120 samples was adopted for final analysis. Fig. 2.4 plots the median of signal strength of different access points at all of the 48 points, which are identified by their distance to the location of the Netgear access point.

2.4.2 LOESS local regression on training set

We utilize the statistical computing package **R** [14] to perform LOESS local regression on the training set. The degree of regression function is $d = 2$ and the value of smoothing parameter q is determined based on AIC metric as described in Section 2.3.3.3. Fig. 2.5 shows the LOESS regression results with 95% confidence interval of three access point built upon the training set. Comparing the curves shown in this figure with those in Fig. 2.4, we can see that the LOESS model built upon the training set fitted will with the signal strength features of each access point.

2.4.3 Maximum likelihood estimation on test set

We apply MLE to the test set and estimate the location of each test point. Estimation accuracy is evaluated by the difference between the estimated locations and the true ones. Fig. 2.6 plots the likelihood function of signal strength received at point $18.5m$, where the estimated location is determined by the peak of the likelihood curve. In this figure, the peak of the likelihood curve occurs at location $18.3m$, so we obtain the estimated location of this point and can further derive the estimation error for this point as $18.5 - 18.3 = 0.2m$.

In Fig. 2.7, we give out the estimation results for all the test points with 95% bootstrapping confidence intervals. From this figure, we can see that the confidence interval for point $18.486m$ is $[15.682m, 21.543m]$. That is, we have a 95% confidence to say that the true location of this point is in this range $[15.682m, 21.543m]$.

We further plot the experimental cumulative distribution function (CDF) of MLE error of our method in Fig. 2.8. From the figure, it is easy to know that 25% of test points have a location error around $0.7m$, while 50% with error around $1.7m$ and 75% with error around $2.5m$. We compare the location error of our method with that of others and show the results in Table 2.1. From the table, we can see that the estimation accuracy of our method is higher than that of RADAR [3] by 40%∼60% and Lim's approach [11] by 30%.

2.4.4 Impact of changing the size of training set

It is not surprising that increasing the size of training set, i.e., choosing more points to build the radio map, can reduce estimation error, because the LOESS model is built from more information about the environment. However, our experimental results do show that the performance of our method is not affected by the changes of the size of training set very much.

Table 2.1 **Comparison of one-dimensional location error**

	25%	**50%**	**75%**
RADAR	$1.92m$	$2.94m$	$4.69m$
Lim's method	$0.97m$	$2.57m$	$3.56m$
Our method	$0.69m$	$1.71m$	$2.53m$

We change the size of training set by adjusting the distance between the training points. For example, if we reduce the distance from $2m$ to $1m$, the size of training set will be doubled. To study the impact of changing the size of training set on estimation accuracy, we show in Fig. 2.9 that how the mean of MLE error vary as a function of the distance. Fig. 2.9 shows that changing the distance from $1m$ to 6 m, that is, reducing the size of training set to $\frac{1}{6}$, only increases the MLE error by 25% (from $2m$ to $2.5m$). It implies that our method is very robust to the changes of the size of training set. Meanwhile, in this scenario, we can save 83% of the time for collecting the training data (we do not show the time data here). So, for our method we can choose a relatively small size of training set as long as the level of estimation error is acceptable.

2.4.5 Impact of changing the number of access points

Another factor affecting the performance of our method is the number of access points. If we deploy more access points, we can expect more accurate estimations, because the LOESS model can exploit more information to build the radio map and estimate locations.

Fig. 2.10 shows the mean of MLE error as a function of the number of access points. It illustrates how we can improve estimation accuracy by increasing the number of access points. For example, Fig. 2.10 shows that we can reduce the mean of MLE error by 20% (from $2m$ to $1.6m$) if we deploy 4 extra access points (i.e., increasing the number of access points from 2 to 6). However, the improvement on estimation accuracy becomes less when there are sufficient access points. For instance, even after we increase the number of access points from 12 to 20, we can only reduce the mean of MLE error by at most 10%. The example implies that deploying too many access points is not necessary.

2.4.6 Setup for two-dimensional experiments

Besides the one-dimensional experiments, we also tested our method in a simple two-dimensional scenario. The two-dimensional experiments were conducted in Room 319 of Snedecor Hall at Iowa State University. Fig. 2.11 illustrates the experimental setup, where

two access points (Belkin F5D 6130 and Netgear WGR614) were placed at the two sides of the room. We selected 6 positions (Position $1 \sim 6$) as the training set for building the radio map. As shown in Fig. 2.11, these 6 positions were 4 meters apart from each other and form two squares. At each position, we collected signal strength for about 3 minutes and generate 180 samples. Then, we selected 50 (stable) samples out of the 180 ones for further analysis. Our test set contained only two points (Test 1 & 2) that were near the center of each square. At each test point, we collected signal strengths for about one minute and generated 10 samples.

When collecting signal strength, we considered the following precautions:

1. Data are collected in the middle of the night to minimize the environmental noise (human activities, etc).

2. The antennas of both access points and the laptop's wireless adapter were pointed vertically at all time to minimize the signal variation caused by antenna orientation and signal multipath.

3. The examiners stood fairly far from the laptop after pushing the start button in order to minimize the multipath interference from a human.

2.4.7 Maximum likelihood estimation on test points in a two-dimensional scenario

Similar to the one-dimensional experiments, in our two-dimensional experiments, we fitted a local regression model on the training set and estimated the location of test points using MLE. In our test setup, we set the coordinates of Position 1 as (0, 0), where the x axis is from right to left and the y axis is from top to bottom.

Fig. 2.12 show a log likelihood plot of test point Test 1, whose coordinates are $(2.0, 2.0)$. In the figure, the darkest point indicates the maximum log likelihood, which is hence the estimated location for the test point. Fig. 2.12(a) plots the log likelihood over the entire region (the entire room) and Fig. 2.12(b) shows the log-likelihood in the region surrounding the lowest signal level point, whose coordinates were $(2.092, 2.715)$.

Table 2.2 shown the true location, estimated location, and location error for two test points. The experimental results show that our localization method has good performance even in the two-dimensional scenario with location errors as low as $1.55m$.

Table 2.2 **Two-dimensional estimation results**

Point	True location	Estimation	Error
1	(2.0, 2.0)	(2.092, 2.715)	$0.72m$
2	(5.0, 1.9)	(6.552, 1.882)	$1.55m$

2.5 Conclusions and future work

We propose a statistical indoor localization method for supporting location-based access control (LBAC). Our method uses LOESS regression to build the distribution of WLAN signal strength and estimates locations using the MLE method. In addition, our method estimates the 95% confidence interval by utilizing the Bootstrapping module. Compared with others, our method is simpler and provides a higher accuracy. It does not need to in incorporate with any physical model for indoor signal strength and can even produce a meaningful confidence interval for its estimation. The experimental results show that the estimation error of our method is less than $2m$. Hence, it can better support LBAC applications than the other localization approaches we analyzed.

Our method can be applied to two-dimensional or three-dimensional localization with almost no changes. We tested our method in a simple two-dimensional scenario and we plan to conduct more experiments on two-dimensional and three-dimensional estimation in the future. We will also study how to utilize real-time information updates to our LOESS model.

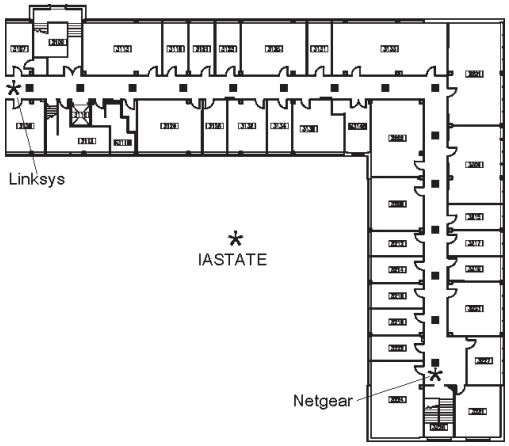


Figure 2.2 The floor plan of Coover Hall for one-dimensional experiments, where the stars are access points and the black blocks represent the training and test points.

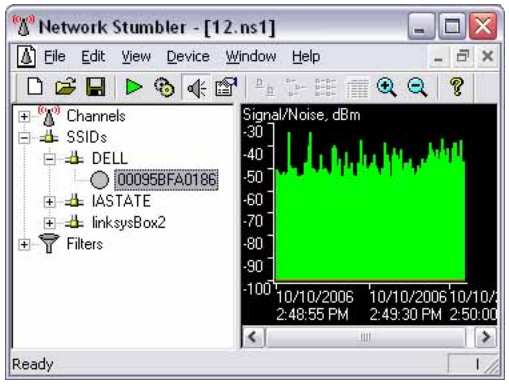
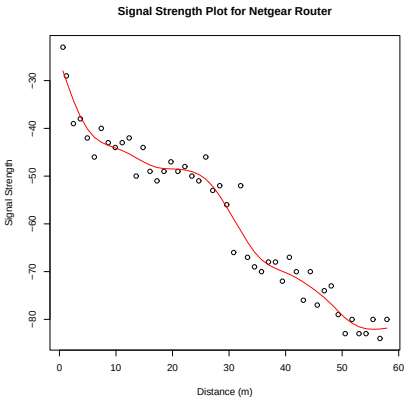
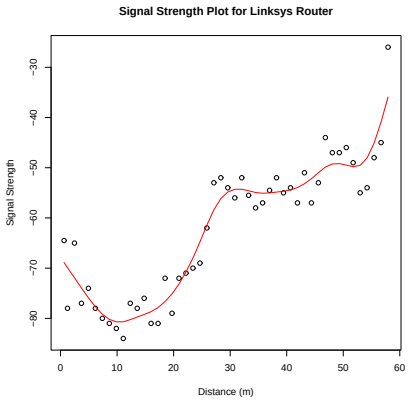


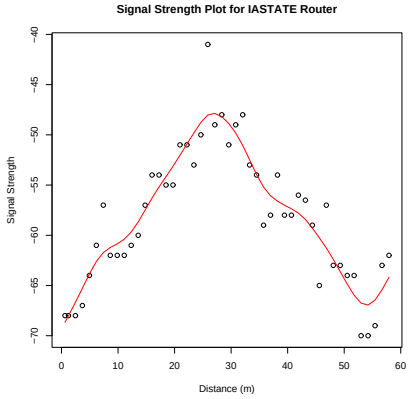
Figure 2.3 A snapshot of NetStumbler



(a) Netgear

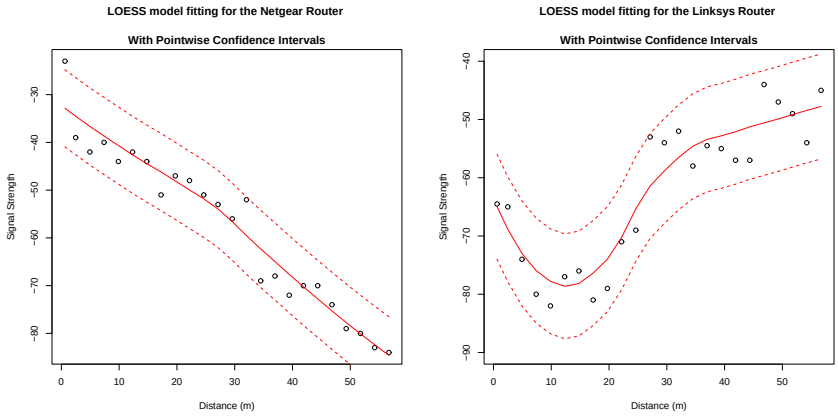


(b) Linksys

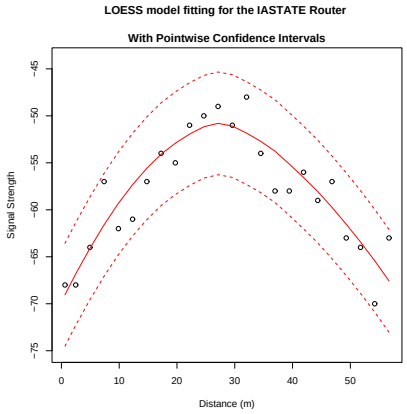


(c) IASTATE

Figure 2.4 The signal strength of three access points measured at different points



(a) Netgear (b) Linksys



(c) IASTATE

Figure 2.5 The LOESS regression models of three access points with 95% confidence interval

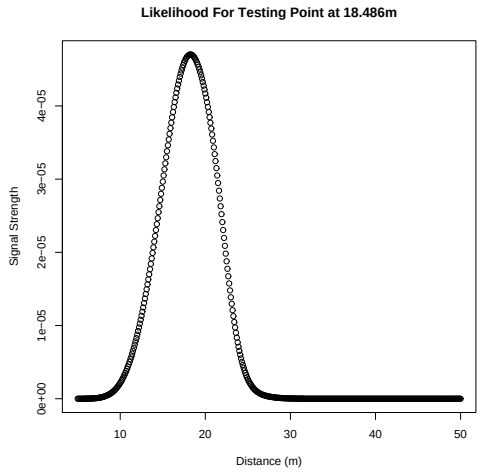


Figure 2.6 The likelihood function for the point at $18.486m$

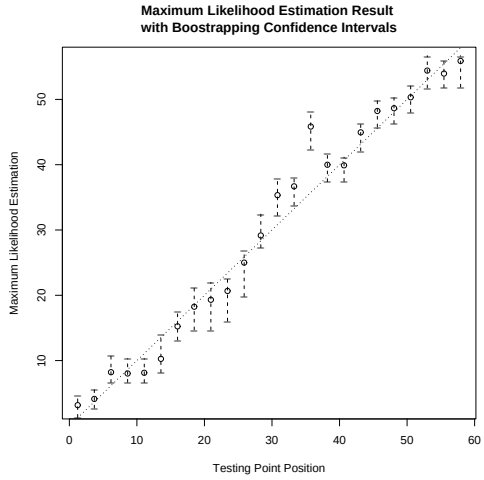


Figure 2.7 The MLE results with 95% bootstrapping confidence intervals

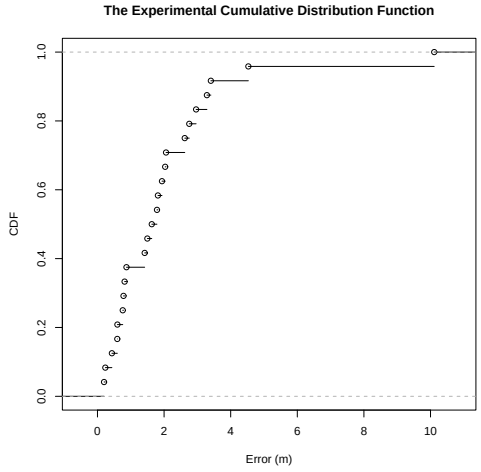


Figure 2.8 The Cumulative Distribution Function (CDF) of MLE error

Figure 2.9 The mean of MLE error as a function of the distance between training points

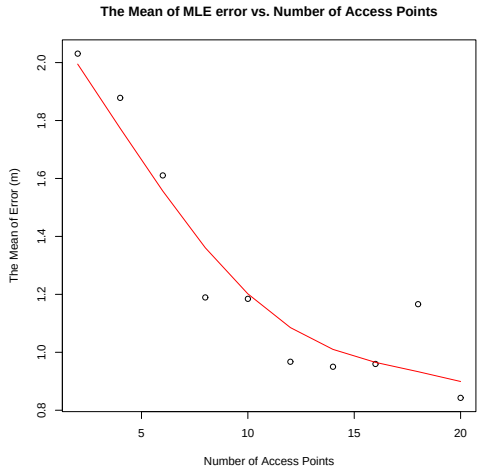


Figure 2.10 The mean of MLE error as a function of the number of access points

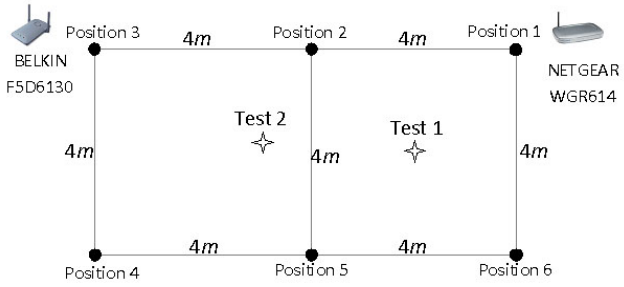
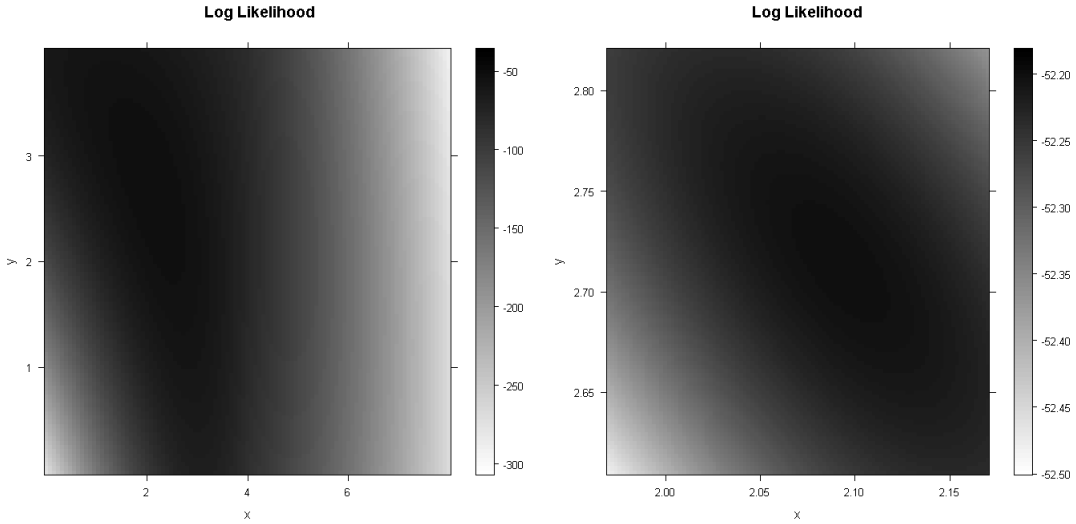


Figure 2.11 The setup for two-dimensional experiments at Snedecor Hall 319, Iowa State University



(a) Entire region

(b) The region surrounding point (2.092, 2.715) with the maximum log likelihood -52.20

Figure 2.12 The log likelihood for point Test 1 at location (2.0, 2.0), where (a) is the plot for the entire region and (b) is the plot for the region surrounding the darkest point at (2.092, 2.715) that has the maximum log likelihood -52.20

APPENDIX A. Selections Of R Code

A.1 Basics Functions

```
#calculate likelihood, result will be used on optimize
LL <- function(P, S, Los)
{
  Result <- 0
  for(AP in 1:length(APs))
  {
    TheLo <- Los[[AP]]
    ThePredict <- predict(TheLo, newdata=P, se=TRUE)
    #the fitted value
    m <- ThePredict$fit
    #the se
    se <- ThePredict$se
    TheS <- TheLo$s
    TheSd <- sqrt(TheS^2+se^2)
    Result <- Result + log(dnorm(S[AP], mean=m, sd=TheSd))
  }
  return(Result)
}
```

```
#want to write a function to find the maximum likelihood
MLE1 <- function(S, Range=c(0.5,60), Los=PartLo.List.1  ,TheAcu=0.001)
{
  return(optimize(LL, Range, tol=TheAcu, maximum=TRUE,S=S, Los = Los ))
}

loess.aic <- function (x) {
        if (!(inherits(x,"loess"))) stop("Error: argument must be a loess object
        # extract values from loess object
        span <- x$pars$span
        n <- x$n
        traceL <- x$trace.hat
        sigma2 <- sum( x$residuals^2 ) / (n-1)
        delta1 <- x$one.delta
        delta2 <- x$two.delta
        enp <- x$enp

        aicc <- log(sigma2) + 1 + 2* (2*(traceL+1)) / (n-traceL-2)
        #aicc1<- n*log(sigma2) + n* ((delta1/(delta2*(n+enp)))/(delta1^2/delta2)
        aicc1<- n*log(sigma2) + n* ((delta1/delta2)*(n+enp)/(delta1^2/delta2)-2
        gcv  <- n*sigma2 / (n-traceL)^2

        result <- list(span=span, aicc=aicc, aicc1=aicc1, gcv=gcv)
        return(result)
}
bestLoess <- function(model, criterion=c("aicc", "aicc1", "gcv"), spans=c(.5, .9

    criterion <- match.arg(criterion)
```

```
    f <- function(span) {

        mod <- update(model, span=span)
        loess.aic(mod)[[criterion]]
        }
    result <- optimize(f, spans)
    list(span=result$minimum, criterion=result$objective)      }

LogLikelihood.1 <- function(TheLo, S, Range=c(5,50), By = 0.1)
{
  NewP <- seq(Range[1],Range[2],by=By)
  ThePredict <- predict(TheLo, NewP, se=TRUE)

  #the fitted value
  m <- ThePredict$fit
  #the se
  se <- ThePredict$se
  TheS <- TheLo$s
  TheSd <- sqrt(TheS^2+se^2)

  return(log(dnorm(S,mean=m,sd=TheSd)))
}
```

A.2 MLE Functions

```
PartLo.List.1 <- NULL

#calculate the loess model for the four half data
for( i in 1:4)
```

```
{
  Data <- DataList.1[[i]]
  Data.Sub <- Data[sub.1,]
  templo <- loess(Signal~Position, data=Data.Sub)
  blo <- bestLoess(templo)
  templo <- loess(Signal~Position, data=Data.Sub, span=blo$span)
  PartLo.List.1 <- c(PartLo.List.1, list(templo))
}

names(PartLo.List.1) <- Names



By <- 0.1
TestS <- 8
Range <- c(1,50)
NewP <- seq(Range[1],Range[2],by=By)
likelihood <- rep(0,length(NewP))
No.AP <- c(4)
for( i in No.AP)
{
  #the plot to identify the data

  templ <- LogLikelihood.1(PartLo.List.1[[i]],DataList.1[[i]]$Signal[TestS],Rang
  plot(templ~NewP, ylim=c(-10,0))
  likelihood <- likelihood + templ
}

plot(likelihood~NewP)
```

```
DataList.1[[1]]$Position[TestS]
P1 <- NewP[which.max(likelihood)]

#calculate the maximum likelihood position more acurately
By <- 0.01
Range <- c(P1-1,P1+1)
NewP <- seq(Range[1],Range[2],by=By)
likelihood <- rep(0,length(NewP))
for( i in No.AP)
{
  #the plot to identify the data

  templ <- LogLikelihood.1(PartLo.List.1[[i]],DataList.1[[i]]$Signal[TestS], Ran
  plot(templ~NewP)
  likelihood <- likelihood + templ
}

plot(likelihood~NewP)

DataList.1[[1]]$Position[TestS]
NewP[which.max(likelihood)]
```

A.3 Bootstrap Code

```
#the bootstrapping confidence interval
#paramatric bootstrapping
#the number of bootstrapping
NoB <- 2000
```

```
#need to do 4 for all 4 access points
BootSignal <- NULL
for( i in 1:4)
{
  a <- predict(PartLo.List.1[[i]], se = TRUE)
  m <- a$fit
  the.sd <- PartLo.List.1[[i]]$s
  tempBoot <- matrix(rnorm(length(m)*NoB,mean=m,sd=the.sd),byrow=TRUE, nrow=NoB)
  BootSignal <- c(BootSignal, list(tempBoot))
}

Position <- Median.1[[1]]$Position[sub.1]
BootResult <- NULL
APs <- c(1,2,3,4)
for(BNO in 1:2000)
{

  lo.boot <- NULL
  for( i in 1:4)
  {
    Signal <- BootSignal[[i]][BNO,]
    templo <- loess(Signal~Position)
    blo <- bestLoess(templo)
    templo <- loess(Signal~Position, span=blo$span)

    lo.boot <- c(lo.boot, list(templo))
  }
```

```
TheResult <- NULL

for(TestP in sub.2)
{
  By <- 0.1
  Range <- c(1,59)
  NewP <- seq(Range[1],Range[2],by=By)
  likelihood <- rep(0,length(NewP))


  for( AP in APs)
  {
    templ <- LogLikelihood.1(lo.boot[[AP]],Median.1[[AP]]$Signal[TestP],Range=
    #plot(templ~NewP, ylim=c(-10,0))
    likelihood <- likelihood + templ
  }
  #find the maximum positon
  #make it more acurate
  P1 <- NewP[which.max(likelihood)]
  By <- 0.01
  Range <- c(P1-0.1,P1+0.1)
  NewP <- seq(Range[1],Range[2],by=By)
  likelihood <- rep(0,length(NewP))
  for( AP in APs)
  {
    templ <- LogLikelihood.1(lo.boot[[AP]],Median.1[[AP]]$Signal[TestP], Range
    #plot(templ~NewP)
```

```
      likelihood <- likelihood + templ
    }
    #plot(likelihood~NewP)
    TheResult <- c(TheResult, NewP[which.max(likelihood)])
  }
  BootResult <- rbind(BootResult, TheResult)
}
save(BootSignal, BootResult, file="c:\\BootResult.3.dat")
TheBootResultQuantile <- matrix(rep(0,3*24),ncol=3)
for( i in 1:24)
{
  TheBootResultQuantile[i,] <- quantile(BootResult[,i], probs=c(0.025,0.5,0.975)

}
plot(RealP, r.1234[,2])

for(i in 1:24)
{
  points( c(RealP[i], RealP[i]),  c(TheBootResultQuantile[i,1],TheBootResultQua
  lty=2,type="o", pch="_")
}
points(RealP, TheBootResultQuantile[,1], pch="-")
points(RealP, TheBootResultQuantile[,3], pch="-")
points(RealP, RealP, pch=5)
plot(RealP ~ RealP, pch=5)

#the CDF of the errors of fitting by 10 points
#I should try
```

```
plot(ecdf(abs(RealP - r.1234[,2])))
```

A.4 WINBUG Code

```
#need AP.no, SS, EE, x.data, m.data, sigma.data, signal
inputPath <- "Y:/LocationSensing/NetStumblerData/2006.8.23.lib.ThirdFloor/TXT/"
SignalStrengthOffset <- -149

#a function to get the signal date base

APs <- c("DELL", "linksysBox2")
PlotFlag <- TRUE

#the function to plot the signal strength
PlotS <- function(NameIndex, APIndex)
{
  InputFileName <- paste(inputPath, as.character(NameIndex), ".out.txt", sep="")
  r <- read.table(InputFileName, header=F)
  colnames(r) <- c("name", "MAC", "Hour", "Minutes", "Second", "Signal")
  r$Signal <- r$Signal + SignalStrengthOffset

  r.part <- subset(r, name==APs[APIndex])
  print(nrow(r.part))
  plot(r.part$Signal, type="o")
}

PlotS(21,1)

#the function come from multiple APs, it read in the data
```

```
SignalInput <- function(No, TheAPs, DeleteOutliers=TRUE, PLOT=FALSE)
{

  FileIndex <- Index$I[No]
  FileName <- paste(inputPath, as.character(FileIndex), ".out.txt", sep="")
  r <- read.table(FileName, header=F)
  colnames(r) <- c("name", "MAC", "Hour", "Minutes", "Second", "Signal")
  r$Signal <- r$Signal + SignalStrengthOffset
  r <- r[Index$S[No]:Index$E[No], ]

  L <- length(TheAPs)
  Result <- NULL
  if(DeleteOutliers)
  {
    for( i in 1:L)
    {
      temp <- r[r$name==TheAPs[i],]
      #the boxplot result
      bp <- boxplot(temp$Signal,plot=FALSE)
      upper <- bp$stats[5,1]
      lower <- bp$stats[1,1]
      #get rid of the outliers
      temp <- temp[temp$Signal > lower & temp$Signal < upper,]
      #put the temp into result
      Result <- rbind(Result,temp)

    }
  }
```

```
  else
  {
    Result <- r
  }
  if(PLOT)
  {
    plot(Result$Signal, type="o")
  }

  return (Result)
}

TheIndex <- seq(42, 546, by=42)
TheIndex <- c(21, TheIndex, 561)
L <- length(TheIndex)
S <- rep(0,L)
E <- rep(0,L)
Index <- data.frame(TheIndex, S, E)
colnames(Index) <- c("I", "S", "E")

Index[1,2:3] <- c(51,150)
Index[2,2:3] <- c(150,249)
Index[3,2:3] <- c(231,330)
Index[4,2:3] <- c(80,179)
Index[5,2:3] <- c(51,150)
Index[6,2:3] <- c(51,150)
Index[7,2:3] <- c(51,150)
Index[8,2:3] <- c(51,150)
```

```
Index[9,2:3] <- c(1,100)
Index[10,2:3] <- c(71,170)
Index[11,2:3] <- c(1,100)
Index[12,2:3] <- c(51,150)
Index[13,2:3] <- c(41,140)
Index[14,2:3] <- c(51,150)
Index[15,2:3] <- c(51,150)

#k <- 15
#A <- 1
#PlotS(Index$I[k], A)

Info <- data.frame(as.character(TheIndex),TheIndex, row.names=NULL)

#skip should be done after input the data Info <- Info[c(1:(skipped-1), (skipped

L <- nrow(Info)
TheN <- 100
TotalData <- NULL
R<- NULL
for(i in 1:L)
{
  r <- SignalInput( i, APs, PLOT=PlotFlag)
  r["Distance"] <- Info[i,2]
  r["Group"] <- Info[i,1]
  TotalData <- rbind(TotalData,r)

  m.dell <- mean(r[r$name=="DELL",]$Signal)
```

```
  sigma.dell <- sd(r[r$name=="DELL",]$Signal)
  md.dell <- median(r[r$name=="DELL",]$Signal)

  m.linksys <- mean(r[r$name=="linksysBox2",]$Signal)
  sigma.linksys <- sd(r[r$name=="linksysBox2",]$Signal)
  md.linksys <- median(r[r$name=="linksysBox2",]$Signal)

  md <- median(r$Signal)
  temp <- c(Info[i,2],m.dell, md.dell, sigma.dell, m.linksys, md.linksys, sigma.
  R <- rbind(R,temp)
}
NewPoints <- seq(21,561, by=1)
R <- data.frame(R, row.names=NULL)
colnames(R) <- c("Distance", "M.dell", "Md.dell", "sigma.dell", "M.linksys", "Md
#without skip
#Dell

Ratio <- 1.005/21

R$Distance <- R$Distance*Ratio
NewPoints <- NewPoints*Ratio
TotalData$Distance <- TotalData$Distance*Ratio

dell.md.lo <- loess(Md.dell~Distance, data=R, span=0.4)
dell.md.predic <- predict(dell.md.lo, newdata=NewPoints, se=T)
plot(Signal~Distance, data=TotalData, subset= (name=="DELL"), xlab="Distance(m)"
points(dell.md.predic$fit~NewPoints, type="l")
points(Md.dell~Distance, data=R, col="red", pch=4, cex=3)
```

```
#Linksys

linksys.md.lo <- loess(Md.linksys~Distance, data=R, span=0.4)
linksys.md.predic <- predict(linksys.md.lo, newdata=NewPoints, se=T)

plot(Signal~Distance, data=TotalData, subset= (name=="linksysBox2"), xlab="Dista
points(M.linksys~ Distance , data= R, col="red", pch=4, cex=3)
points(linksys.md.predic$fit~NewPoints, type="l")

skipped <- 3
R.sub <- R[c(1:(skipped-1), (skipped+1):nrow(R)),]
#plot of dell

dell.md.lo <- loess(Md.dell~Distance, data=R.sub, span=0.4)
dell.md.predic <- predict(dell.md.lo, newdata=NewPoints, se=T)
plot(Signal~Distance, data=TotalData, subset= (name=="DELL"))
points(dell.md.predic$fit~NewPoints, type="l")
points(Md.dell~Distance, data=R, col="red", pch=4, cex=3)
#plot of linksys
#the loess fit for the linksys data
linksys.md.lo <- loess(Md.linksys~Distance, data=R.sub, span=0.4)
linksys.md.predic <- predict(linksys.md.lo, newdata=NewPoints, se=T)

plot(Signal~Distance, data=TotalData, subset= (name=="linksysBox2"))
points(M.linksys~ Distance , data= R, col="red", pch=4, cex=3)
points(linksys.md.predic$fit~NewPoints, type="l")
```

```
#use the maximum likelihood to estimate the position

Diff.x <- NULL

for(skipped in 2:14)
{
  R.sub <- R[c(1:(skipped-1), (skipped+1):nrow(R)),]
  dell.md.lo <- loess(Md.dell~Distance, data=R.sub, span=0.4)
  dell.md.predic <- predict(dell.md.lo, newdata=NewPoints, se=T)
  linksys.md.lo <- loess(Md.linksys~Distance, data=R.sub, span=0.4)
  linksys.md.predic <- predict(linksys.md.lo, newdata=NewPoints, se=T)
  x.data <- NewPoints
  m.data <- rbind(dell.md.predic$fit, linksys.md.predic$fit)
  sigma.data <- rbind(dell.md.predic$se.fit, linksys.md.predic$se.fit)
  signal <- R[skipped,c(3,6)]
  signal <- c(signal$Md.dell,signal$Md.linksys)
  ll <- NULL
  for(i in 1:length(NewPoints))
  {
    thex <- x.data[i]
    thel <- 1
    for( j in 1:length(signal) )
    {
      temp.m <- m.data[j, i]
      temp.sigma <- sigma.data[j,i]
      temp.s <- signal[j]
      temp.d <- dnorm(temp.s, mean=temp.m, sd=temp.sigma)
      thel <- thel*temp.d
```

```
    }
    ll <- rbind(ll,c(thex, thel))
  }
  ll <- data.frame(ll)
  colnames(ll) <- c("x", "L")
  plot(L~x, data=ll, main="Likelihood Plot")
  ind <- which.max(ll$L)
  Diff.x <- rbind(Diff.x,c(R$Distance[skipped], x.data[ind]))
}

#change the unit to meters

Diff.x <- data.frame(Diff.x)

colnames(Diff.x) <- c("x","xp")
Diff.x["Diff"] <- Diff.x$xp- Diff.x$x
plot(xp~x, data=Diff.x, main="Estimation Plot")
plot(Diff~x,data=Diff.x)
plot( Diff.x$Diff,type="s",data=Diff.x)
plot(ecdf(Diff.x$Diff))
#the winbugs code part
#need AP.no, SS, EE, x.data, m.data, sigma.data, signal
AP.No <- length(APs)
EE <- length(NewPoints)
x.data <- NewPoints
m.data <- rbind(dell.md.predic$fit, linksys.md.predic$fit)
sigma.data <- rbind(dell.md.predic$se.fit, linksys.md.predic$se.fit)
signal <- R[skipped,c(3,6)]
```

```
signal <- c(signal$Md.dell,signal$Md.linksys)

#the whole problem is for the median estimate

path <- "Y:/LocationSensing/NetStumblerData/2006.8.23.lib.ThirdFloor/R/"
BugFileName <- "bugscode1.bug"
BugFile <- paste(path, BugFileName, sep="")

DataList <- list("AP.No", "EE", "x.data", "m.data", "sigma.data", "signal")
parameters <- c ("x")

IN <- function()
{
  list(xx=300)
}

BugsResult <- bugs(DataList,  inits=IN, parameters, BugFile, n.chains=1, debug=F
x <- BugsResult$sims.list$x
hist(x)

summary(BugsResult)

#calculate of likelihood
ll <- NULL
for(i in 1:length(NewPoints))
{
  thex <- x.data[i]
  thel <- 1
```

```
  for( j in 1:length(signal) )
  {
    temp.m <- m.data[j, i]
    temp.sigma <- sigma.data[j,i]
    temp.s <- signal[j]
    temp.d <- dnorm(temp.s, mean=temp.m, sd=temp.sigma)
    thel <- thel*temp.d

  }
  ll <- rbind(ll,c(thex, thel))
}
ll <- data.frame(ll)
colnames(ll) <- c("x", "L")
plot(L~x, data=ll)
which.max(ll$L)
```

BIBLIOGRAPHY

[1] H. Akaike. A new look at the statistical model identification. *Automatic Control, IEEE Transactions on*, 19(6):716–723, 1974.

[2] Claudio A. Ardagna, Marco Cremonini, Ernesto Damiani, Sabrina De Capitani di Vimercati, and Pierangela Samarati. Supporting location-based conditions in access control policies. In *ASIACCS '06: Proceedings of the 2006 ACM Symposium on Information, computer and communications security*, pages 212–222, New York, NY, USA, 2006. ACM.

[3] Paramvir Bahl and Venkata N. Padmanabhan. Radar: An in-building RF-based user location and tracking system. In *INFOCOM (2)*, pages 775–784, 2000.

[4] W. S. Cleveland. Robust locally weighted regression and smoothing scatterplots. *Journal of the American Statistical Association*, (74):829–836, 1979.

[5] William S. Cleveland and Susan J. Devlin. Locally weighted regression: An approach to regression analysis by local fitting. *Journal of the American Statistical Association*, 83(403):596–610, 1988.

[6] Michael J. Covington, Wende Long, Srividhya Srinivasan, Anind K. Dev, Mustaque Ahamad, and Gregory D. Abowd. Securing context-aware applications using environment roles. In *SACMAT '01: Proceedings of the sixth ACM symposium on Access control models and technologies*, pages 10–20, New York, NY, USA, 2001. ACM.

[7] Dorothy E. Denning and Peter F. MacDoran. Location-based authentication: grounding cyberspace for better security. pages 167–174, 1998.

[8] R. A. Fisher. On the mathematical foundations of theoretical statistics. *Philosophical Transactions of the Royal Society of London. Series A, Containing Papers of a Mathematical or Physical Character*, 222:309–368, 1922.

[9] Peter Hall. *The Bootstrap and Edgeworth Expansion*. Springer Series in Statistics. Springer, New York, 1992.

[10] Andy Harter, Andy Hopper, Pete Steggles, Andy Ward, and Paul Webster. The anatomy of a context-aware application. In *In Mobile Computing and Networking*, pages 59–68. ACM Press, 1999.

[11] H. Lim, L.-C. Kung, J. C. Hou, and H. Luo. Zero-configuration, robust indoor localization: Theory and experimentation. In *INFOCOM 2006. 25th IEEE International Conference on Computer Communications. Proceedings*, pages 1–12, April 2006.

[12] Dragoş Niculescu and Badri Nath. Vor base stations for indoor 802.11 positioning. In *MobiCom '04: Proceedings of the 10th annual international conference on Mobile computing and networking*, pages 58–69, New York, NY, USA, 2004. ACM.

[13] Nissanka B. Priyantha, Anit Chakraborty, and Hari Balakrishnan. The cricket location-support system. In *MobiCom '00: Proceedings of the 6th annual international conference on Mobile computing and networking*, pages 32–43, New York, NY, USA, 2000. ACM Press.

[14] R Development Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2009. ISBN 3-900051-07-0.

[15] Roy Want, Andy Hopper, Veronica Falc ao, and Jonathan Gibbons. The active badge location system. *ACM Trans. Inf. Syst.*, 10(1):91–102, 1992.

[16] Moustafa Youssef and Ashok Agrawala. The horus wlan location determination system. In *MobiSys '05: Proceedings of the 3rd international conference on Mobile systems, applications, and services*, pages 205–218, New York, NY, USA, 2005. ACM.

[17] Moustafa Youssef, Adel Youssef, Chuck Rieger, Udaya Shankar, and Ashok Agrawala. Pinpoint: An asynchronous time-based location determination system. In *MobiSys '06: Proceedings of the 4th international conference on Mobile systems, applications and services*, pages 165–176, New York, NY, USA, 2006. ACM.